In the Claims:

Please amend the claims as follows:

1. (Currently Amended) A method for maximizing CPU performance in a multiprocessor comprising:

configuring a computer system with multiple processors with a common physical memory address space accessible by all processors, and non-shared memory local to a processor;

organizing data elements stored in a shared resource designed to support data manipulation functions, wherein said shared resource is stored in shared memory; and

pipelining execution of an operation with software instructions by partitioning the operating into a series of sequential steps, said instructions comprising:

executing write operations in ~~local~~ memory local to a processor in an arbitrary order and at any time prior to storing a pointer from an existing element of said shared resource stored on a computer readable medium to a new element of said shared resource, wherein said pointer is stored in said shared resource;

explicitly indicating a set of write operations to non-local memory be conducted in a specified order; and

~~forcing execution of~~ executing said write operations to non-local memory prior to ~~precede said~~ storing said pointer from said existing element of said shared resource to said new element of said shared resource in response to said indicating.

2. (Previously Presented) The method of claim 1, further comprising assigning first and second registers of a CPU for storing associated first and second instruction addresses.

3. (Previously Presented) The method of claim 2, further comprising providing a third instruction referencing said registers.

4. (Original) The method of claim 3, wherein said third instruction specifies ordering between said first and second instructions.

5. (Original) The method of claim 4, wherein said third instruction indicates said first instruction's execution attaining a first specified state of execution prior to said second instruction's execution attaining a second specified state of execution.

6. (Original) The method of claim 5, wherein said first and said second specified states of execution are selected from the group consisting of: committing instruction execution, initiating memory access, completing a memory access, initiating an I/O access, completing an I/O access, and completing instruction execution.

7. (Previously Presented) The method of claim 1, further comprising assigning a sequence number to an associated instruction for maintaining instruction ordering.

8. (Previously Presented) The method of claim 7, further comprising statically encoding said sequence number within said instruction.

9. (Previously Presented) The method of claim 7, further comprising dynamically encoding said sequence number within said instruction.

10. (Previously Presented) The method of claim 1, further comprising placing a range of instructions into a hierarchical ordering system.

11. (Previously Presented) The method of claim 10, further comprising implementing a special instruction for maintaining a hierarchical execution of said instruction.

12. (Currently Amended) A computer system comprising:

multiple processors with a common physical memory address space accessible by all processors, and non-shared memory local to a processor;

~~a processor in communication with shared storage media;~~

data elements organized and stored in a shared resource ~~said storage media having data elements organized in a shared resource~~ designed to support data manipulation functions; and

compiler directives to execute an operation partitioned into a series of sequential steps, comprising:

a first instruction in communication with said processor and said shared resource for executing write operations in ~~local~~ memory local to a processor in an arbitrary order; and

a second instruction in communication with said processor and said shared resource for explicitly indicating a set of write operations to non-local memory to be conducted in a specified order; and

a special instruction in communication with said processor and said shared

resource to force execution of said write operations to non-local memory to precede storage of a pointer from an existing element of said shared resource to a new element of said shared resource, wherein said pointer is stored in said shared resource.

13. (Original) The processor of claim 12, further comprising a first register to store a first instruction address and a second register to store a second instruction address.

14. (Previously Presented) The processor of claim 13, further comprising a third instruction to manage order of execution of said first and second instructions.

15. (Original) The processor of claim 14, wherein execution of said second instruction is responsive to said first instruction reaching a specified state of execution.

16. (Original) The processor of claim 15, wherein said specified state of execution is selected from the group consisting of: committing instruction execution, initiating memory access, completing a memory access, initiating an I/O access, completing an I/O access, and completing instruction execution.

17. (Original) The processor of claim 12, wherein said first and second instructions are assigned a sequence number to specify an order of instruction execution.

18. (Original) The processor of claim 17, wherein said sequence number is statically encoded within said instruction.

19. (Original) The processor of claim 17, wherein said sequence number is dynamically encoded within said instruction.

20. (Previously Presented) The processor of claim 12, further comprising a manager to place a range of instructions in a hierarchical order.

21. (Previously Presented) The processor of claim 19, further comprising a special instruction to maintain execution of said instruction in said hierarchical order.

22. (Currently Amended) A computer system, comprising:
multiple processors with a common physical memory address space accessible by all

processors, and non-shared memory local to a processor;

~~a processor in communication with shared storage media;~~

~~said storage media having~~ data elements organized in a shared resource designed to support data manipulation functions; and

partition of an operation into a series of sequential steps, comprising:

a first instruction in communication with said processor and said shared resource ~~for allowing~~ to allow write operations in ~~local~~ non-shared memory to occur in an arbitrary order;

a second instruction in communication with said processor and said shared resource ~~for~~ to explicitly indicating a set of write operations to ~~non-local~~ shared memory to be conducted in a specified order, wherein write operations to ~~non-local~~ shared memory must execute prior to storage of a pointer from an existing element of a shared resource into a new element of said shared resource; and

a third instruction in communication with said processor and said shared resource ~~for managing~~ to manager order of execution of said first and second instructions;

wherein execution of said second instruction is responsive to said first instruction reaching a specified state of execution and said specified state of execution is selected from the group consisting of: committing instruction execution, initiating memory access, completing a memory access, initiating an I/O access, completing an I/O access, and completing instruction execution.

23. (Original) The processor of claim 22, a first register to store a first instruction address and a second register to store a second instruction address.

24. (Original) The processor of claim 22, wherein said first and second instructions are assigned a sequence number to specify an order of instruction execution.

25. (Currently Amended) The processor of claim 22, further comprising a special instruction in communication with said processor and said shared resource to maintain execution of said instructions in said hierarchical order.